

# A Parallel Way to Select the Parameters of SVM Based on the Ant Optimization Algorithm

Chao Zhang  
Software Engineering  
Software College  
SiChuan University,China  
Email: zhch040200@gmail.com

Hong-Cen Mei  
Software Engineering  
Software College  
SiChuan University,China  
Email: hongcenmei@yeah.net

Hao Yang  
Software Engineering  
Software College  
SiChuan University,China  
Email: chongqingyanghao@yeah.net

**Abstract**—A large number of experimental data shows that Support Vector Machine (SVM) algorithm has obvious a large advantages in text classification, handwriting recognition, image classification, bioinformatics, and some other fields. To some degree, the optimization of SVM depends on its kernel function and Slack variable, the determinant of which is its parameters  $\delta$  and  $c$  in the classification function. That is to say, to optimize the SVM algorithm, the optimization of the two parameters play a huge role. Ant Colony Optimization (ACO) is optimization algorithm which simulate ants to find the optimal path. In the available literature, we mix the ACO algorithm and Parallel algorithm together to find a well parameters.

**Keyword:** SVM, Parameters, ACO, OpenCL, Parallel

## I. SUPPORT VECTOR CLASSIFICATION AND PARAMETERS

SVM is based on the principle of structural risk minimization, using limited training samples to obtain the higher generalization ability of decision function. Suppose a sample set  $(x_i, y_i)$ , where  $i = 1, 2, \dots, N$  means the number of training samples,  $x \in R$  means the sample characteristics,  $y \in \{+1, -1\}$  means the sample classification.

SVM Classification function:

$$y = \omega x + b \quad (\omega \text{ means weight vector, } b \text{ means setover})$$

Functional margin :

$$\gamma_1 = y(x + b) = yf(x)$$

functional margin is the minimum margin from hyperplane  $(\omega, b)$  to  $T(x_i, y_i)$

Geometrical margin:

$$\gamma_2 = \frac{y(\omega x + b)}{\|\omega\|} = \frac{|f(x)|}{\|\omega\|}$$

When classifying a data point, the larger the margin is, the more credible the classification is. So to improve the credibility is to maximize the margin. The  $\omega$  and  $b$  can be proportional scaled through the functional margin, thus the value of  $y = \omega x + b$  can be any large. Such that it is not appropriate to maximize a value. But in the geometrical margin, when scaling the  $\omega$  and  $b$ , the value of  $\gamma_2$  can't be change. So it is appropriate to maximize a value.

Slack variable:

$\varepsilon_i > 0$  Using to allow the data to deviate from the hyperplane

to a certain extent.

Radial Basis Function kernel:

$$k(x_1, x_2) = e^{-\frac{\|x_1 - x_2\|^2}{2\sigma^2}}$$

Maximum margin classifier:

$$MAX \gamma_2 \quad s.t. \gamma_{1i} \geq \gamma_1 - \varepsilon_i, i = 1, 2, 3, \dots, n$$

Let:  $\gamma_1 = 1$ , so

$$MAX \frac{1}{\|\omega\|} + C \sum_{i=1}^n \varepsilon_i s.t. \gamma_{1i} \geq \gamma_1 - \varepsilon_i, i = 1, 2, 3, \dots, n$$

The constraints is associated with objective function through the Lagrange function:

$$L(\omega, b, \alpha) = \frac{1}{2} \|\omega\|^2 - \sum_{i=1}^n \alpha_i (y_i ((\omega x + b) - 1) + C \sum_{i=1}^n \varepsilon_i$$

Let:

$$\Theta(\omega) = MAX L(\omega, b, \alpha)$$

When all the constraints is contented, the  $\Theta(\omega) = \frac{1}{2} \|\omega\|^2$  is the value that we first want to minimized. So objective function:

$$MIN \Theta(\omega) = MIN MAX L(\omega, b, \alpha)$$

Dual function:

$$MIN \Theta(\omega) = MAX MIN L(\omega, b, \alpha)$$

To solve the problem, requiring:

$$\frac{\partial L}{\partial \omega} = 0 \Rightarrow \omega = \sum_{i=1}^n \alpha_i x_i y_i$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{i=1}^n \alpha_i y_i = 0$$

$$\frac{\partial L}{\partial \varepsilon_i} = 0 \quad \text{means} \quad C - \alpha_i - \gamma_i = 0, i = 1, 2, \dots, n$$

$$MAX MIN L(\omega, b, \alpha) = \frac{1}{2} \|\omega\|^2 - \sum_{i=1}^n \alpha_i (y_i ((\omega x + b) - 1)$$

$$= MAX \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j k(x_i, x_j)$$

The minimized duel problem:

$$\begin{aligned} \text{MAX} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j k \langle x_1, x \rangle \\ \text{s.t. } 0 \leq \alpha_i \leq C, i = 1, 2, 3 \dots n; \sum_{i=1}^n \alpha_i y_i = 0 \end{aligned}$$

The final classification function:

$$\begin{aligned} y &= \omega x + b \\ \sum_{i=1}^n \alpha_i y_i k \langle x_1, x \rangle + b \end{aligned}$$

## II. MODIFIED ANT COLONY OPTIMIZATION ALGORITHM

Different from traditional problem of TCP, in this algorithm, the coordinate is used to represent the node. In a two-dimensional rectangular coordinate system, the significance of X is defined as the significant digit of parameters C and  $\delta$ , Y is varied from 0 to 10.[1] The significant digit of C are assumed to be the five, and the highest level of C is hundred's place. Similarly, assume that the significant digit of  $\delta$  are also assumed to be the five, and the highest level of  $\delta$  is The Unit. To realize the ant colony optimization, we follow the steps below:

- 1) Suppose there are  $m$  ants. Each ant  $k(k = 1 \sim m)$  has a one-dimensional array  $Path(k)$  which has  $n$  elements ( $n$  is the total significant digit of C and  $\delta$ ), its used to store the vertical coordinates  $y$  of each point which the ant  $k$  visited.
- 2) Set the loop time  $N=0$  to  $N_{max}$ . Initialize the each points pheromone concentration  $\tau(x, y) = \tau_0(x = 0 \sim n, y = 0 \sim 9)$ . Set  $x = 1$ .
- 3) Set  $y = 1$  to  $n$ . Calculate the deflection probability  $P_k$  of each ant move to the node vertical line  $L_x$ . Then select the next point via roulette wheel and store the points vertical coordinates  $y$  to  $Path_k$

$$\begin{aligned} P_k(x, y) &= \frac{\tau^\alpha(x, y) \eta^\beta(x, y)}{\sum_{j=0}^9 \tau^\alpha(x, y) \eta^\beta(x, y)} \\ \tau(x, y) &= \rho \tau(x, y) + \Delta \tau(x, y) \\ \Delta \tau(x, y) &= \sum_{k=1}^m \Delta \tau_k(x, y) \\ \Delta \tau(x, y) &= \begin{cases} \frac{Q}{1 - Acc_k} & \text{,if ant } k \text{ visited the point}(x, y) \\ 0 & \text{,others} \end{cases} \end{aligned}$$

The  $Acc_k$  mean the ant  $k$  s accuracy of cross-validation

$P_k$  means the deflection probability point  $(x - 1, y)$  to  $(x, y)$

Q is a constant

W: the weight coefficient;

Set: the minimum acceptable accuracy;

- 4) Let  $x = x + 1$ , if  $x \leq n$ , turn to step 3; else turn to step 5.
- 5) Record this motion path  $Path_k$  calculate the mapping data  $(c, \delta)$ .

- 6) Make the training samples evenly divided into  $k$  mutually exclusive subsets of  $S_1, S_2, \dots, S_k$ ;
- 7) Calculate the cross-validation accuracy:
  - a) Initialize  $i=1$ ;
  - b) Make the  $S_i$  subset reserved for test sets, and set the rest as the training set, training SVM;
  - c) Calculate the  $i^{th}$  subsets Sample Classification Accuracy  $Acc_i$ , set  $i = i + 1$ , When  $i < k + 1$ , repeat the step b);
  - d) Calculate the mean of  $k$  Sample Classification Accuracy  $\hat{Acc}$ :

$$Acc = \frac{Right}{Error + Right}$$

$$\hat{Acc} = \frac{\sum_{i=1}^k Acc_i}{k}$$

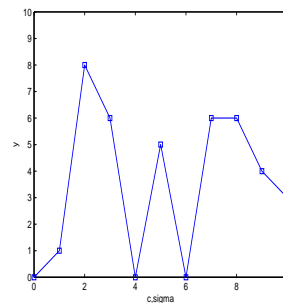
Right: Correctly classified (+1) number of samples

Error: Misclassification (-1) number of samples

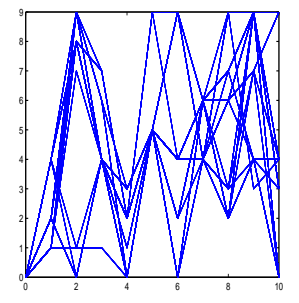
$\hat{Q}$ : the mean of Sample Classification Accuracy

- 8) Update the pheromone concentration  $\tau(x, y)$  at each points by  $\hat{Acc}$ . Clear  $Path(k_i)$ ;
- 9) Reset  $N = N + 1$ . When  $N \leq N_{max}$ , and the entire colony has not converged to follow the same path, then turn to step 3;  $N \leq N_{max}$ , and the entire colony substantially converged to follow the same path, then the algorithm ends. Take the last update of the path and its mapping data to  $(c, \delta)$ , that is the SVM parameters C,  $\delta$  final optimization results.

In order to validate the algorithm, we do simulation experiment on matlab R2010b and PC with windows7 64-bit operating system, 4 G memory, core i5 processor. We divide the *wine\_SVM* data set into a sample set and a test set. Then we will have 90 training datas and 88 test datas. We set ACO parameter  $m=30$ ,  $N=500$ ,  $\rho=0.7$ ,  $Q=100$ ,  $\alpha=1$ ,  $\beta=1$ . In order to calculate the classification accuracy, we introduced the LIBSVM.[3]. LIBSVM has gained wide popularity in machine learning and many other areas.[2]



(a) The best accuracy's path



(b) All accuracy's paths

The best accuracy is 95.4545% and the convergence accuracy is 86.3636% (76/88).

The parameter  $c$ : 18.605, parameter  $\sigma$ : 0.6643

## III. PARALLEL OPTIMIZE THE PARAMETERS

The Open Computing Language, is an open specification for heterogeneous computing released by the Khronos Group2 in 2008. It resembles the NVIDIA CUDA3 platform, but can

be considered as a superset of the latter, they basically differ in the following points[4]:

- OpenCL is an open specification that is managed by a set of distinct representatives from industry, software development, academia and so forth.
- OpenCL is meant to be implemented by any compute device vendor, whether they produce CPUs, GPUs, hybrid processors, or other accelerators such as digital signal processors (DSP) and field-programmable gate arrays (FPGA).
- OpenCL is portable across architectures, meaning that a parallel code written in OpenCL is guaranteed to correctly run on every other support device.

In this ant colony optimization algorithm, we have  $m$  ants and we loop it  $N_{max}$  times. If we give  $m$  a large number, such as ten thousand or one hundred thousand, our update of node pheromone concentration will be more accurate and reliable, but we will spend more time. As we all know, every cycle, each ant's access to nodes are unrelated with others, so we can parallel the ant access process. In this article, we use openCL to realize the parallel of ant[4].

---

OpenCL kernel for the ant-based solution construction

---

```

global_size = number_ants;
visited[global_size × number_nodes] = {-1};

for ( i=0 to n-1) do
{
    sum_prob = 0;
    for (j=0 to 9) do
    {
        selection_prob[global_id × number_nodes + j]
        = choice_info[i × n + j];
        Sum_prob = sum_prob +
        selection_prob[global_id × number_nodes + j];
    }
    j=0;
    p = selection_prob[global_id × number_nodes + j];
    while p < random(0, sum_prob) do
    {
        j=j+1;
        p = p + selection_prob[global_id × number_nodes + j];
    }
    visited[global_id × number_nodes + i] = j;
}

```

---

Training samples are divided into subset\_number subset on average and each subset has sample\_number samples. One subset( $S_i$ ) see as test set and the other subsets ( $S_{1\sim i}, S_{i\sim k}$ ) see as a training set(each subset has c samples), then according to the current parameters ( $c, \delta$ ) training the SVM, calculating error of K-fold cross validation.

---

OpenCL kernel for sample classification accuracy

---

```

group_size = subset_number;
local_size = sample_number;

```

```

for (i=0 to group_size - 1)
{
    Right = Error = Q = 0;
    for(j = 0 to local_size - 1)
    {
        f(x) = sign( $\sum_{i=1}^n \alpha_i y_i k(\text{group\_id}, \text{local\_id}) + b$ );
        if f(x) = 1
            Right ++;
        else
            Error ++;
    }
    Q = Q +  $\frac{\text{Right}}{\text{Right} + \text{Error}}$ ;
}
Q = Q / group_size;

```

---

#### IV. CONCLUSION

Through the ant colony optimization algorithm, we can find a satisfactory parameter of SVM, and the convergence accuracy can be guaranteed more than 85%. There are also many other ways to optimize parameters, such as Genetic algorithm (GA)[5], dynamic encoding algorithm [6] for handwritten digit recognition, Particle swarm optimization(PSO)[7]. We also can parallel Ant Colony Optimization, article [8] introduced a new way which parallel Ant Colony Optimization on Graphics Processing Units. Article [9] improving ant colony optimization algorithm for data clustering.

#### ACKNOWLEDGMENT

#### REFERENCES

- [1] P. F. LIU Chun-bo, WANG Xian-fang, "Parameters selection and stimulation of support vector machines based on ant colony optimization algorithm," *J. Cent. South Univ.*, 2008.
- [2] C. chung Chang and C.-J. Lin, "Libsvm : a library for support vector machines," *Linux Journal*, 2001.
- [3] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," vol. 2, pp. 1–27, 2011.
- [4] E. by Helio J.C. Barbosa, *Ant Colony Optimization - Techniques and Applications*. InTech, Chapters published February 20, 2013 under CC BY 3.0 license, ISBN 978-953-51-1001-9, 203 pages.
- [5] J. L.-c. ZHENG Chun-hong, "Automatic parameters selection for SVM based on GA[C]," *NJ:IEEE Press*, 2004:1869-1872.
- [6] Y. Park, S.-W. Kim, and H.-S. Ahn, "Support vector machine parameter tuning using dynamic encoding algorithm for handwritten digit recognition," 2005.
- [7] X. Li, S. dong Yang, and J. xun Qi, "A new support vector machine optimized by improved particle swarm optimization and its application," *Journal of Central South University of Technology*, vol. 13, pp. 568–572, 2006.
- [8] A. Delevacq, P. Delisle, and M. Gravel, "Parallel Ant Colony Optimization on Graphics Processing Units," *Journal of Parallel and Distributed Computing*, vol. 73, 2013.
- [9] R. Tiwari, M. Husain, S. Gupta, and A. Srivastava, "Improving ant colony optimization algorithm for data clustering," pp. 529–534, 2010.